

Modularisierung in der Praxis

change this!



Conways Gesetz:

Conways Gesetz:

Die Struktur der Software spiegelt die Struktur der Organisation wider.

Definition

Schnittstelle (allgemein): Teil eines Systems, das dem Austausch von Informationen, Energie oder Materie mit anderen Systemen dient.

– *Wikipedia, die freie Enzyklopädie*

Definition

Schnittstelle (allgemein): Teil eines Systems, das dem Austausch von Informationen, Energie oder Materie mit anderen Systemen dient.

– *Wikipedia, die freie Enzyklopädie*

Definition

Schnittstelle: Menge der Annahmen, die das Modul über seine Umgebung macht, sowie jene, die die Umgebung über das Modul macht.

Schnittstellen

Definition

Schnittstelle (allgemein): Teil eines Systems, das dem Austausch von Informationen, Energie oder Materie mit anderen Systemen dient.

– *Wikipedia, die freie Enzyklopädie*

Definition

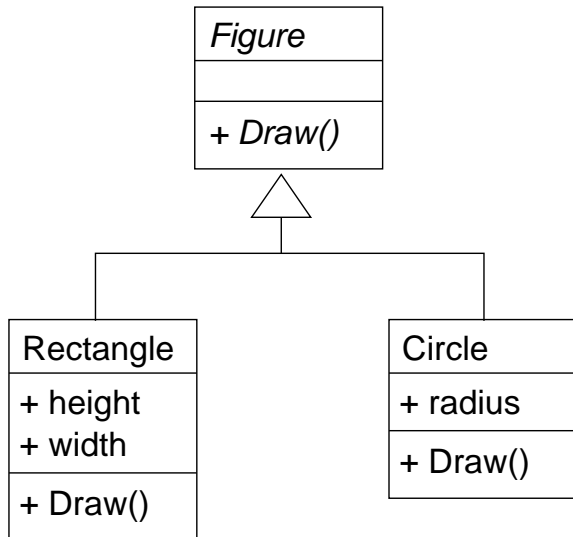
Schnittstelle: Menge der Annahmen, die das Modul über seine Umgebung macht, sowie jene, die die Umgebung über das Modul macht.

Definition

Syntaktische Schnittstelle: öffentliche Attribute und Methoden mit deren Signaturen.

Zwei Module sind über ihre Schnittstellen verbunden und damit abhängig.

Ein Wort zu Schnittstellen



Eine offene Schnittstelle

```
abstract class Figure {  
  
    public abstract void Draw ();  
}  
class Circle extends Figure {  
  
    public void Draw () {};  
    public int radius;  
}  
class Rectangle extends Figure {  
  
    public void Draw () {};  
    public int height;  
    public int width;  
}
```

Geheimnisprinzip (Information Hiding) nach Parnas (1972)

Schnittstellen sind ein Kontrakt zwischen:

- Verwender:
 - darf sich nur auf zugesicherte Annahmen verlassen
 - muss Vorbedingungen einhalten
- Anbieter:
 - muss zugesichertes Verhalten implementieren
 - darf sich nur auf zugesicherte Vorbedingungen verlassen

Geheimnisprinzip (Information Hiding) nach Parnas (1972)

Schnittstellen sind ein Kontrakt zwischen:

- Verwender:
 - darf sich nur auf zugesicherte Annahmen verlassen
 - muss Vorbedingungen einhalten
- Anbieter:
 - muss zugesichertes Verhalten implementieren
 - darf sich nur auf zugesicherte Vorbedingungen verlassen

Der Kontrakt führt zu einer Kopplung zwischen Verwender und Anbieter.

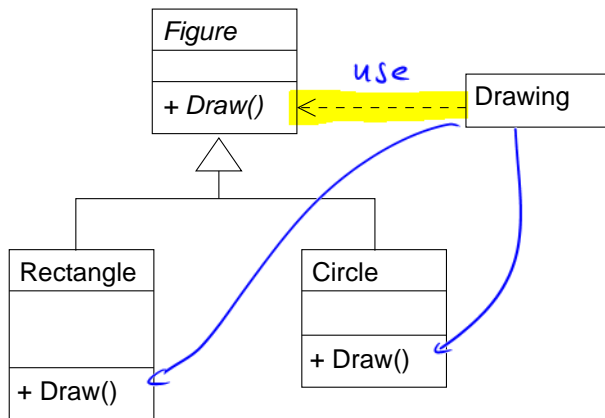
Schnittstellen werden so entworfen, dass

- Kopplung auf das Mindestmaß beschränkt wird;
- d.h. die Details, die sich ändern können, werden hinter Schnittstelle verborgen.

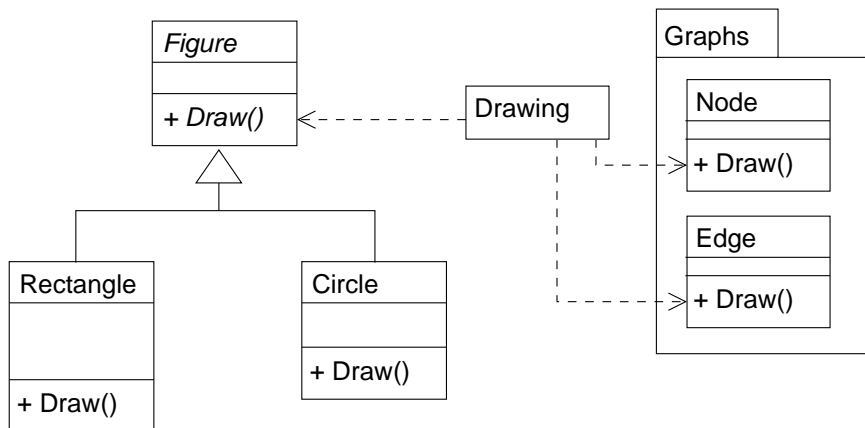
```
abstract class Figure {  
  
    public abstract void Draw ();  
}  
class Circle extends Figure {  
  
    public void Draw () {};  
    private int radius;  
}  
class Rectangle extends Figure {  
  
    public void Draw () {};  
    private int height;  
    private int width;  
}
```

```
public void Drawing (Figure A_Figure, int Times) {  
    for (int i = 0; i < Times; i++) {  
        A_Figure.Draw ();  
    }  
}
```

Klassendiagramm

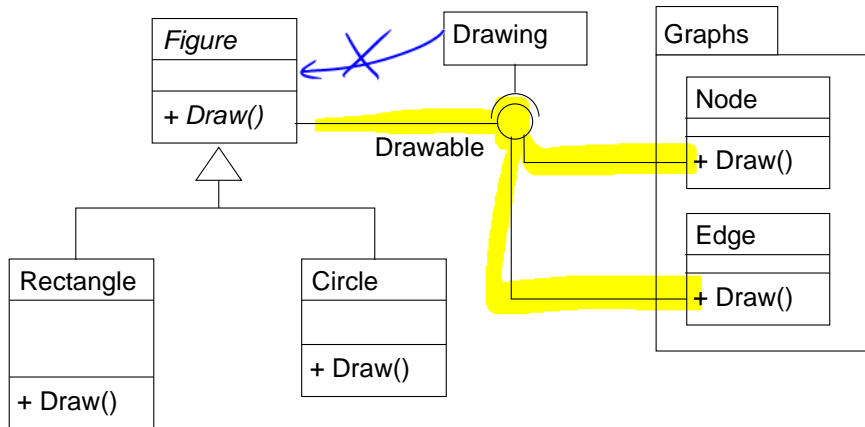


Klassendiagramm



```
public void Drawing (Figure A_Figure , int Times) {  
    for (int i = 0; i < Times; i++) {  
        A_Figure.Draw ();  
    }  
}
```

Klassendiagramm



Schnittstelle als eigenes Sprachkonstrukt

Schnittstelle:

```
interface Drawable {  
    void Draw ();  
}
```


Schnittstelle als eigenes Sprachkonstrukt

Schnittstelle:

```
interface Drawable {  
    void Draw ();  
}
```

Schnittstellenverwender:

```
public void Drawing (Drawable Drawable_Object, int Times) {  
    for (int i = 0; i < Times; i++) {  
        Drawable_Object.Draw ();  
    }  
}
```

Schnittstelle als eigenes Sprachkonstrukt

Schnittstelle:

```
interface Drawable {  
    void Draw ();  
}
```

Schnittstellenverwender:

```
public void Drawing (Drawable Drawable_Object, int Times) {  
  
    for (int i = 0; i < Times; i++) {  
        Drawable_Object.Draw ();  
    }  
}
```

Schnittstellenanbieter:

```
abstract class Figure implements Drawable {}
```

- Schnittstelle ist Kontrakt zwischen Anbieter und Verwender, der die erlaubten wechselseitigen Annahmen festlegt
- Programmiersprachen erlauben die Spezifikation syntaktischer Eigenschaften von Schnittstellen
- moderne Programmiersprachen bieten Schnittstellen als eigenes Sprachkonstrukt an
- nur wenige erlauben die Spezifikation semantischer Eigenschaften
 - Vor- und Nachbedingungen
 - weitere Zusicherungen, wie z.B. Speicher- und Zeitkomplexität

Definition

Kopplung: Grad der Abhängigkeit zwischen Modulen.

Kopplung und Zusammenhalt

Definition

Kopplung: Grad der Abhängigkeit zwischen Modulen.

Definition

Zusammenhalt (Kohärenz): Verwandtschaft der Teile eines Moduls.

Kopplung und Zusammenhalt

Definition

Kopplung: Grad der Abhängigkeit zwischen Modulen.

Definition

Zusammenhalt (Kohärenz): Verwandtschaft der Teile eines Moduls.



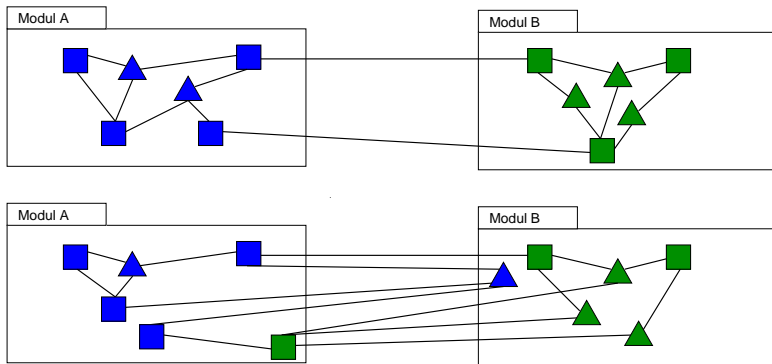
Kopplung und Zusammenhalt

Definition

Kopplung: Grad der Abhängigkeit zwischen Modulen.

Definition

Zusammenhalt (Kohärenz): Verwandtschaft der Teile eines Moduls.



Stufen des Zusammenhalts (von schlecht nach gut)

| Stufe | innerhalb einer Funktion/Moduls | betrifft |
|--------------------------|---|--------------------|
| kein Zusammenhalt | rein zufällige Zusammenstellung | Module |
| Ähnlichkeit | z.B. ähnlicher Zweck, also etwa alle Matrixoperationen; auch Fehlerbehandlung | Module |
| zeitliche Nähe | Verwendung zum selben Zeitpunkt, z.B. Initialisierung, Abschluss | Module, Funktionen |
| gemeinsame Daten | Zugriff auf bestimmte Daten, exklusiv, z.B. Kalenderpaket (Systemuhr) | Funktionen, Module |
| Hersteller / Verbraucher | ein Teil erzeugt, was der andere verwendet | Module, Funktionen |
| einziges Datum | Kapselung einer Datenstruktur, Zusammenfassung der Operationen | Module, Funktionen |
| einzigste Operation | Operation, die nicht mehr zerlegbar ist | Funktionen |

Stufen der Kopplung (von schlecht nach gut)

| Stufe | zwischen Funktionen/Modulen | betrifft |
|-------------------|--|------------------------|
| Einbruch | Veränderung des Codes | Funktionen |
| volle Öffnung | Zugriff auf alle Daten, z.B. auf alle Attribute einer Klasse | (Module) Funktionen |
| selektive Öffnung | bestimmte Attribute sind zugänglich oder global durch expliziten Export/Import | Module, Funktionen |
| Funktionskopplung | nur Funktionsaufruf und Parameter | Module (Funktion) |
| keine Kopplung | Es besteht keine logische Beziehung (Zugriff ist syntaktisch unmöglich) | Module |

Jeder Entwurf ist ein Kompromiss.

- Geringe Kopplung zwischen allen Modulen
- Hoher Zusammenhalt in allen Funktionen und Modulen
- Kriterium der naiven Suche: Jede Einheit sollte einen leicht erkennbaren Sinn haben.

Jeder Entwurf ist ein Kompromiss.

- Geringe Kopplung zwischen allen Modulen
- Hoher Zusammenhalt in allen Funktionen und Modulen
- Kriterium der naiven Suche: Jede Einheit sollte einen leicht erkennbaren Sinn haben.
- Abstraktion: Implementierungsdetails verbergen
- Kapselung von Dingen, die sich ändern können (Geheimnisprinzip; Information Hiding)

Jeder Entwurf ist ein Kompromiss.

- Geringe Kopplung zwischen allen Modulen
- Hoher Zusammenhalt in allen Funktionen und Modulen
- Kriterium der naiven Suche: Jede Einheit sollte einen leicht erkennbaren Sinn haben.
- Abstraktion: Implementierungsdetails verbergen
- Kapselung von Dingen, die sich ändern können (Geheimnisprinzip; Information Hiding)
- Etwa gleich große Einheiten (Module, Funktionen). Abweichungen sollten plausibel sein; generell dürfen einfache Objekte größer sein als komplizierte.

Kriterien für einen guten Software-Entwurf (Forts.)

- Uniforme Entwurfsstrategie; wenn z.B. eine Schichtenstruktur gewählt wurde, sollte sie nicht an irgendeiner Stelle korrumpiert sein.
- Uniforme Gestaltung der Schnittstellen.
- Uniforme Benennung.

¹Nein, nicht *der* Michael Jackson.

Kriterien für einen guten Software-Entwurf (Forts.)

- Uniforme Entwurfsstrategie; wenn z.B. eine Schichtenstruktur gewählt wurde, sollte sie nicht an irgendeiner Stelle korrumpiert sein.
- Uniforme Gestaltung der Schnittstellen.
- Uniforme Benennung.
- Prinzip der Isomorphie zur Realität (Michael Jackson¹): Die Software sollte der Realität strukturell gleichen, damit die Änderungen der Realität in der Software leicht nachvollzogen werden können.

¹Nein, nicht *der* Michael Jackson.

Software Architecture Analysis Method (SAAM) (Kazman u. a. 1996):

- ① Lege wichtige Qualitätsaspekte fest
- ② Beschreibe alternative Modularisierungen
- ③ Entwickle Szenarien für die Qualitätsaspekte
- ④ Spiele die Szenarien durch und bewerte die Modularisierungen
- ⑤ Betrachte Wechselwirkungen zwischen den Qualitätsaspekten
- ⑥ Fasse die Evaluation zusammen

Beispiel: Key Word in Context (KWIC) (Parnas 1972)

westfälischer Friede

Friede von Osnabrück

Bistum Osnabrück

Beispiel: Key Word in Context (KWIC) (Parnas 1972)

zyklische Vertauschung
jeder Zeile



| |
|----------------------|
| westfälischer Friede |
| Friede von Osnabrück |
| Bistum Osnabrück |

| |
|----------------------|
| westfälischer Friede |
| Friede westfälischer |
| Friede von Osnabrück |
| Osnabrück Friede von |
| von Osnabrück Friede |
| Bistum Osnabrück |
| Osnabrück Bistum |

Beispiel: Key Word in Context (KWIC) (Parnas 1972)

zyklische Vertauschung
jeder Zeile



| |
|----------------------|
| westfälischer Friede |
| Friede von Osnabrück |
| Bistum Osnabrück |

Sortierung der
Zeilen



| |
|----------------------|
| westfälischer Friede |
| Friede westfälischer |
| Friede von Osnabrück |
| Osnabrück Friede von |
| von Osnabrück Friede |
| Bistum Osnabrück |
| Osnabrück Bistum |

| |
|----------------------|
| Bistum Osnabrück |
| Friede von Osnabrück |
| Friede westfälischer |
| Osnabrück Bistum |
| Osnabrück Friede von |
| von Osnabrück Friede |
| westfälischer Friede |

- Änderbarkeit
 - Eingabeformat ändert sich
 - größere Dateien müssen verarbeitet werden
 - riesige Dateien müssen verarbeitet werden
- separate Entwickelbarkeit
 - verteile Arbeit an Entwicklungsteams
- Performanz
 - berechne KWIC für FB3-Webseiten

Ablauf

- ① Eingabe der Daten
- ② Interne Speicherung der Daten
- ③ Indizierung (Zeile, Wortanfang, Wortende)

| | |
|---|-------------|
| W e s t f ä l i s c h e r F r i e d e | (1, 1, 13) |
| 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20 | (1, 15, 20) |

| | |
|---|------------|
| B i s t u m O s n a b r ü c k | (2, 1, 6) |
| 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 | (2, 8, 16) |

| | |
|---|------------|
| F r i e d e v o n O s n a b r ü c k | (3, 1, 6) |
| 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20 | (3, 8, 10) |

(Note: In the original image, 'von' is circled in blue, and a blue line connects the end of the third row to the index (3, 12, 20).)

(3, 12, 20)

- ④ Zyklische Rotation der Indizierung
- ⑤ Sortierung der Indizierung
- ⑥ Ausgabe der Indizierung

Erste Modularisierung (ablauforientiert)

